WHAT IS CLAIMED:

1. A method for creating a user-defined type in a database system, comprising:

receiving code that implements a class defining the structure of a user-defined type and methods that can be invoked on instances of the user-defined type;

enforcing a contract against the class, the contract comprising:

a first requirement that the class specify one of a plurality of different formats for persisting instances of the user-defined type in a database store;

a second requirement that the class be capable of returning a null value for the user-defined type; and

a third requirement that the class provide a method for converting the user-defined type to another type; and

enabling instances of the user-defined type to be created only when the class meets the requirements of the contract.

2. The method of claim 1, further comprising storing metadata about the user-defined type for subsequent use by the database system in creating instances of the user-defined type.

3. The method of claim 1, wherein the plurality of different formats for persisting instances of the user-defined type comprises:

a first format in which an instance of the user-defined type is automatically serialized in accordance with a native format of the database system; and

a second format in which an instance of the user-defined type is serialized in a manner defined by the class.

4. The method of claim 3, wherein the plurality of different formats for persisting instances of the user-defined type further comprises a third format in which an instance of the user-defined type is serialized in accordance with a method provided by the MICROSOFT .NET FRAMEWORK.

5. The method of claim 1, wherein the code that implements the class comprises managed code.

6. The method of claim 1, further comprising instantiating the user-defined type as one of a column value in a table, a variable, a parameter of a routine, and a return value of a routine.

7. The method of claim 1, further comprising:

receiving an expression in the query language of the database system, wherein evaluation of the expression requires invocation of a method of an instance of the user-defined type;

translating the expression into a sequence of program code instructions that invoke the required method on the instance of the user-defined type;

invoking the method upon execution of the program code; and

returning a result of the method invocation as the evaluation of the query language expression.

8. The method of claim 7, wherein said method further comprises deserializing the instance of the user-defined type prior to invoking the method on the instance.

9. The method of claim 7, wherein the program code instructions comprise managed code.

10. The method of claim 1, wherein the class that defines the user-defined type comprises a mutator method that, when invoked, enables a value of the user-defined type to be changed, and wherein the method further comprises invoking the mutator method on an instance of the user-defined type to change the value of the instance.

11. The method of claim 10, wherein said step of invoking the mutator method comprises:

deserializing the instance of the user-defined type;

invoking the mutator method to change the value of the instance; and

serializing the instance of the user-defined type having the changed value.

12. The method of claim 1, wherein the class defining the structure and method of the user-defined type further comprises an attribute that specifies that serialized binary representations of instances of the user-defined type will be binary ordered.

13. The method of claim 12, further comprising:

serializing instances of the user defined type such that the binary representations of the instances are binary ordered;

receiving an expression in a query language of the database system that requires the comparison of a first instance of the user defined type to a second instance of the user defined type; and

comparing the serialized binary representations of the first and second instances of the user-defined type to evaluate the expression, without deserializing either instance.

14. The method of claim 12, further comprising:

creating a table in a database store in which a type of a column of the table is defined as the user-defined type; and

creating an index on the column.

15. The method of claim 12, further comprising:

serializing instances of the user defined type such that the binary representations of the instances are binary ordered;

receiving an expression in a query language of the database system, the evaluation of which requires invocation of a method on an instance of the user defined type;

generating a computed column over the expression; and

creating an index over the computed column.

16. The method of claim 1, further comprising:

performing said receiving, enforcing, and enabling steps to create a user-defined type in the context of a first database;

performing said receiving, enforcing, and enabling steps to create a user-defined type in the context of a second database; and

determining whether the user-defined type created in the context of the first database is equivalent to the user-defined type created in the context of the second database.

17. A database system comprising:

a runtime that provides code execution within the database system; and

a database server that receives code that implements a class defining the structure of a user-defined type and methods that can be invoked on instances of the user-defined type and that enforces a contract against the class, the contract comprising:

a first requirement that the class specify one of a plurality of different formats for persisting instances of the user-defined type in a database store;

a second requirement that the class be capable of returning a null value for the user-defined type; and

a third requirement that the class provide a method for converting the user-defined type to another type,

the database server enabling instances of the user-defined type to be created only when the class meets the requirements of the contract.

18. The database system of claim 17, wherein the database server stores metadata about the user-defined type for subsequent use by the database server in creating instances of the user-defined type.

19. The database system of claim 17, wherein the plurality of different formats for persisting instances of the user-defined type comprises:

a first format in which an instance of the user-defined type is automatically serialized in accordance with a native format of the database system; and

a second format in which an instance of the user-defined type is serialized in a manner defined by the class.

20. The database system of claim 19, wherein the plurality of different formats for persisting instances of the user-defined type further comprises a third format in which an instance of the user-defined type is serialized in accordance with a method provided by the MICROSOFT .NET FRAMEWORK.

21. The database system of claim 17, wherein the runtime provides managed code execution within the database system and wherein the code that implements the class comprises managed code.

22. The database system of claim 17, wherein a user of the database system can instantiate the user-defined type as one of a column value in a table, a variable, a parameter of a routine, and a return value of a routine.

23. The database system of claim 17, further comprising a query processor that (i) receives an expression in the query language of the database system, wherein evaluation of the expression requires invocation of a method of an instance of the user-defined type, (ii) translates the expression into a sequence of program code instructions that invoke the required method on the instance of the user-defined type, (iii) invokes the method upon execution of the program code, and (iv) returns a result of the method invocation as the evaluation of the query language expression.

24. The database system of claim 23, wherein the query processor deserializes the instance of the user-defined type prior to invoking the method on the instance.

25. The database system of claim 23, wherein the program code instructions comprise managed code.

26. The database system of claim 17, wherein the class that defines the user-defined type comprises a mutator method that, when invoked, enables a value of the user-defined type to be changed, and wherein the database server invokes the mutator method on an instance of the user-defined type to change the value of the instance.

27. The database system of claim 26, wherein the database server invokes the mutator method by deserializing the instance of the user-defined type, invoking the mutator method to change the value of the instance, and then serializing the instance of the user-defined type having the changed value.

28. The database system of claim 17, wherein the class defining the structure and methods of the user-defined type further comprises an attribute that specifies that serialized binary representations of instances of the user-defined type will be binary ordered.

29. The database system of claim 28, wherein the database server serializes instances of the user defined type such that binary representations of the instances are binary ordered, and

wherein when the database server receives an expression in a query language of the database system that requires the comparison of a first instance of the user defined type to a second instance of the user defined type, the database server compares the serialized binary representations of the first and second instances of the user-defined type to evaluate the expression, without deserializing either instance.

30. The database system recited in claim 28, wherein the database server creates a table in a database store in which a type of a column of the table is defined as the user-defined type, and wherein the database server creates an index on the column.

31. The database system of claim 28, wherein the database server serializes instances of the user defined type such that the binary representations of the instances are binary ordered, and wherein when the database server receives an expression in a query language of the database system, the evaluation of which requires invocation of a method on an instance of the user defined type, the database server generates a computed column over the expression and then creates an index over the computed column.

32. A computer-readable medium having program code stored thereon, the program code, when executed by a computer, causing the computer to perform the steps of the method recited in claim 1.